ASIA-PACIFIC SOLAR RESEARCH CONFERENCE

PEER REVIEWED
ASIA-PACIFIC
SOLAR RESEARCH CONFERENCE

Rui Tang

Generating Residential PV Production and Electricity Consumption Scenarios via Generative Adversarial Networks

Rui Tang[1, 2], Philip H.W. Leong[1], Jonathon Dore[2] and Anthony Vassallo[3]

[1]*Computer Engineering Laboratory, The University of Sydney, Sydney NSW 2006, Australia*
[2]*Solar Analytics Pty Ltd, Sydney NSW 2016, Australia*
[3]*Centre for Sustainable energy development, University of Sydney, Sydney NSW 2006, Australia*
E-mail: rui.tang@sydney.edu.au

Abstract

Due to the natural intermittency of renewable energy resources (such as wind and PV) and high variations in electricity consumption profiles, planning and scheduling optimisations generally require a large amount of historical energy data to produce accurate forecasts. Often in practical situations, historical data is not sufficient to capture the uncertainties in generation and consumption. A promising approach to solve this issue is to generate various generation and consumption scenarios, specifying possible trajectories of solar and load power. Then these scenarios can be incorporated into the optimisation models for infrastructure planning and power operation.

In this paper, we propose a data driven Generative Adversarial Networks (GAN) based model to generate domestic solar production and electricity consumption scenarios. We train our generative model using historical solar and load data collected by Solar Analytics from Australian residential PV customers. Moreover, by using a conditional GAN, we demonstrate we can generate synthetic data conditioned on site specific conditions. By validating the distributions of our generated data against real-time data, we illustrate that we can produce realistic PV generation and consumption profiles.

1. Introduction

In recent years, the residential renewable energy penetration has increased significantly and the trend is expected to continue in the next few decades. Taking rooftop solar in Australia as an example, as of the end of 2017, 17% of electricity generation comes from renewable energy resources where 20.3% of the renewable generation is met by small-scale solar (CEC, 2018). By 2050, 44% of all electricity is expected to be made by residential PV and behind-the-meter batteries (Bloomberg NEF, 2018). As a result, there is a growing interest in planning and scheduling optimisations of residential PV systems especially when integrated energy storage systems are also considered.

One of the greatest challenges in the power system modelling domain is to work with limited amounts of real-time data, especially in the optimisations of storage integrated renewable energy systems where generally a large amount of historical / forecasted generation and consumption data is required. Although the rollouts of smart meters are expected to enable easier data access, most end-consumers, the tools and information are not available (Chandrashekeran, 2018). Moreover, for third parties including installers or researchers, accessing interval data is still difficult due to data protection regulations (e.g. the European Union's General Data Protection Regulation (European Parliament, 2016)) and privacy concerns. As a result, publicly available smart meter datasets are quite limited. Furthermore, due to the intrinsic intermittency of energy resources such as wind or PV and high variations in residential electricity consumptions, a single deterministic

consumption/generation forecast or extrapolation model may be insufficient to capture uncertainties in real-time generation and consumption.

One promising approach to address the above issue is to generate different generation and consumption scenarios, modelling possible trajectories of PV and consumption power. Then we could apply the generated data to optimise system configurations, demand-side management or power scheduling of storage integrated residential renewable energy systems. In the previous studies on power scenario generation, probabilistic models are widely adopted which require some certain statistical assumptions and sampling from fitted probabilistic models (Iversen & Arduin, 2016; Wang & Tanabe, 2016). In recent years, as a type of generative model, Generative Adversarial Networks (GANs) have drawn a lot of attention in computer vision research due to its benefits of generating realistic images using less statistical assumptions and faster runtime without fitting a probabilistic model (Goodfellow et al., 2014; Radford & Chintala, 2016; Arjovsky et al., 2017).

In this paper, we propose a GAN-based tool to generate synthetic residential PV generation and electricity consumption data. The contributions are to:

i. Provide a data driven and robust approach to residential generation and consumption scenarios in which no explicit probabilistic models are required.
ii. Illustrate that the tool can be easily adjusted to generate data conditioned on PV system size, peak electricity consumption or site specific conditions such as consumption patterns.

The remainder of the paper is organized as follows: Section 2 includes a review on GANs and related studies on generating synthetic solar and load data. Section 3 describes the proposed methodology and Section 4 demonstrate our results and a case study. Finally, Section 5 concludes the study and discusses some future work.

2. Literature Review

In previous studies, a few models have been proposed for renewable energy scenario generation. Some studies apply Copula models to construct a multivariate joint distribution using the historical data, then Latin Hypercube Sampling (LHS) or Monte Carlo sampling (MCS) is adopted to generate scenarios based on the obtained multivariate joint distribution (Iversen & Arduin, 2016; Wang & Tanabe, 2016; Tastu et al., 2015). Alternatively, Fourier series and autoregressive moving average (ARMA) models are applied to characterise the seasonality and autocorrelation in the residues of weather or load demand data, then synthetic data is generated using the trained ARMA models (Chen et al., 2017; Suomalainen et al., 2017). Pillai et al. (2014) applied an artificial neural network (ANN) approach to generate aggregated synthetic load data for a region using its typical meteorological year 2 (TMY2) weather data as model inputs. A Wasserstein Generative Adversarial Networks (WGAN) based model is proposed in Chen et al.'s work (2017) to generate synthetic commercial wind and solar generation data.

Generative modelling is a type of machine learning approach which takes samples drawn from a specific data distribution, producing an estimation of that distribution (Goodfellow, 2016). Compared to discriminative models which have made a lot of progress in the last decade (e.g. in image classification), generative models had less of an impact due to the difficulty of approximating many intractable probabilistic computations in maximum likelihood estimation and the difficulty of leveraging the benefits of piecewise linear units in the generative context (Goodfellow et al., 2014). Generative Adversarial Networks (GAN) have drawn a lot of attention in the last few years as they overcome the above difficulties and are able to generate high quality samples without requiring Monte Carlo approximations or Markov chains. In the last couple of years, many modified versions of GANs have been proposed. Conditional Generative Adversarial Networks (CGAN) were proposed by Mirza & Osindero (2014) which allows GAN to generate samples conditioned on class labels. Radford et al. (2015) developed Deep Convolutional Generative Adversarial Networks

(DCGAN) which incorporate Convolutional Neural Network (CNN) structures in a GAN framework and are able to generate high quality images without a multi-stage generation process. Wasserstein Generative Adversarial Networks (WGAN) (Arjovsky et al., 2017) use Wasserstein distance to measure the distances between the generated and real data distributions instead of Jensen–Shannon Divergence adopted in the original GAN model (Goodfellow et al., 2014). As a result, it provides a more meaningful loss metric and better training stability.

Overall, to the best of our knowledge, residential solar generation and load consumption scenario generation has not been previously explored. As GANs seem to be a promising generative approach, we propose a tool which generates synthetic residential load and generation data. Moreover, as variational autoencoders (Kingma & Welling et al., 2013; Rezende et al., 2014) are also one of the most popular generative models, we provide a comparison with GANs in this study.

3. Methodology

3.1. Data Collection

The residential solar and consumption data used in this study is collected from 44 customers via Solar Analytics' smart meters. The dataset includes one year of 5-minute power data from August 2016 to July 2017 for each customer. We set the sampling length of our generative models to one day and apply a random split to divide the dataset into a training set comprising 80% of the data and a test set (20% of the dataset) to evaluate the performance of our generative models. It should be noted the splits are done separately for generation and consumption data.

3.2. Data Normalisation

To accelerate the training processes and to obtain good results in GAN or VAE, it is necessary to normalise our input data to a range between 0 and 1. In this study, we simply divide all the 5-minute solar and load power values by the PV rated DC power and peak load power values respectively for each individual customer. Not only does this provide bounds for our data range, it also allows us to generate new solar and load data based on DC system sizes and peak consumption levels, achieving scale invariance.

3.3. Generative Adversarial Networks (GAN)

The framework of a GAN is presented in Figure 1. Generally, a GAN consists of two artificial neural networks (ANN) based functions, a generator (G) and a discriminator (D). The aim of a generator is to utilise random noise (**z**) to generate samples from a distribution modelling the training set, whereas the goal of the discriminator is to determine which samples are real (drawn from the true data distribution) or fake.

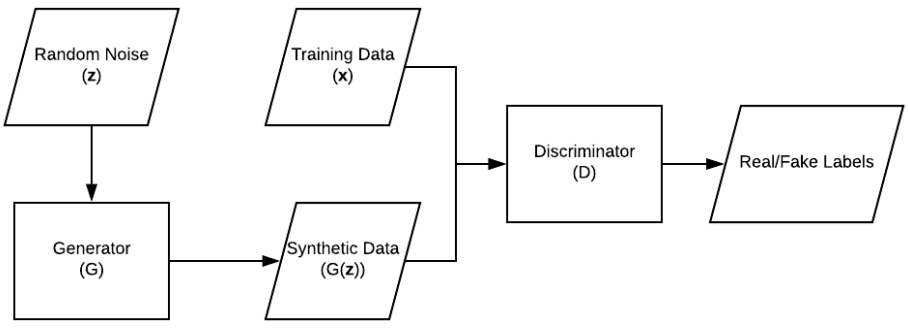

Figure 1. A GAN Framework

The discriminator (D) is essentially a binary classifier with a sigmoid output, its cross-entropy cost can be defined in equation (1):

$$JD\theta D,\ \theta G=-12\ x\sim pdata\log Dx-12\ z\sim pg\log(1-DG(z)) \quad (1)$$

Where $\theta D,\ \theta G$ are parameters of the discriminator and generator, $pdata$ is the targeted data distribution and pg is an estimate of $pdata$ in a generative model. The discriminator attempts to minimise the cost function defined in Equation (1). On the other hand, the generator maximises the chance of the discriminator making a mistake by minimising the cost function shown in Equation (2):

$$JG\theta D,\ \theta G=-12\ z\sim pg\log DG(z) \quad (2)$$

When both generator and discriminator are optimised, pg approaches $pdata$ and the discriminator outputs ~50% fake probability for both real and synthetic datasets.

In this study, we adopt a DCGAN architecture (Radford et al., 2015) with the following adjustments:

- We change the number of filters in each convolutional layer of the discriminator, empirically we found less filters are required for our dataset.
- We use non-square feature maps in our convolutional and de-convolutional layers, this means we could easily adopt our input data dimension (288 data points for a single day) without scaling the input to a $n \times n$ dimension.
- We change the last activation layer of the generator to sigmoid instead of a hyperbolic tangent (tanh) function used in the original GAN and DCGAN papers (Goodfellow et al., 2014; Radford et al., 2015), as our normalised training data is within the range of 0 to 1 and it avoids negative values that occur if tanh is used.

Figure 2 illustrate the DCGAN architecture applied in this study. First a vector of 100 random values from a Gaussian distribution is fed into the generator and 5-minute data for a day is generated using de-convolutional layers. Then the discriminator which includes a few convolutional layers outputs the probabilities on the sample being real/fake.

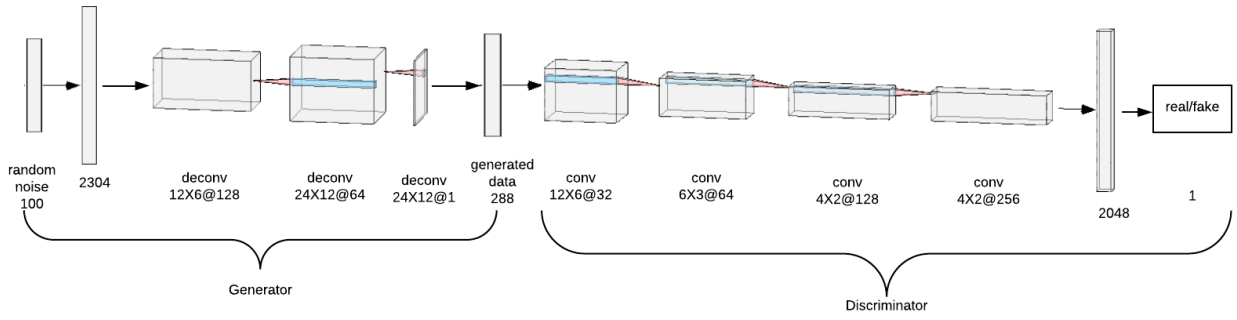

Figure 2. DCGAN Architecture used in this study. Numbers at the bottom of each layer indicate the output dimensions of each layer, a single number describes a dense layer whereas $height \times width$ @ $depth$ show dimensions of a convolutional or de-convolutional layer.

Batch normalisation (Ioffe & Szegedy, 2015) is applied after each convolutional/de-convolutional layer to stabilise the learning process, except for the last generator layer and the first discriminator layer. Rectified linear unit (ReLU) (Nair & Hinton, 2010) and Leaky rectified linear unit (LReLU)

(Maas et al., 2013) activation layers are used for the generator and the discriminator respectively except for their output layers which both use sigmoid functions.

We also adopted the same optimiser settings from the original DCGAN paper, where an Adam optimiser (Kingma & Ba, 2014) is used with learning rate and momentum $\beta 1$ set to be 0.0002 and 0.5. Dropout layers are also added for each convolutional layer in the discriminator to reduce overfitting and the dropout rate is set to be 0.25. The batch size is set to be 32, this means for each training iteration, we randomly select 32 days of real time data to train the DCGAN model.

3.4. Variational Autoencoder (VAE)

A variantional autoencoder (VAE) (Kingma & Welling et al., 2013; Rezende et al., 2014) is another generative model that has been quite popular in recent years. It is a deep Bayesian model which models the relationships between latent variables (**z**) and observed data (**x**). Similar to a standard autoencoder, it consists of an encoder and a decoder which are both ANNs. The goal of the encoder is to derive the latent representation of **x** using **z** which is from a normal prior distribution, hence it models $p\theta \boldsymbol{zx}$ which is the posterior distribution of **z**. After that, the decoder which models $p\theta \boldsymbol{xz}$, maps the latent points back to the original data **x**. The true posterior $p\theta \boldsymbol{zx}$ is analytically intractable therefore a variational posterior $q\phi \boldsymbol{zx}$ is used to approximate it using variational parameters $\mu\phi \boldsymbol{x}$ and $\sigma\phi \boldsymbol{x}$. These two parameters, which can be trained by a neural network in VAE (see inference network in Figure 3), specify the mean and covariance of a multivariate Gaussian distribution that characterises $q\phi \boldsymbol{zx}$. The optimisation objective function of a VAE can be defined by the evidence lower bound (ELBO):

$$ELBOq = q\phi \boldsymbol{zx} \log p\theta \boldsymbol{xz} - KLq\phi \boldsymbol{zx} || p(z) \quad (3)$$

The first term of Equation (3) measures the reconstruction loss which pushes the decoded samples to match the original inputs. The second term $KLq\phi \boldsymbol{zx} || p(z)$ is used as a regularisation term which measures the Kullback–Leibler (KL) divergence between $q\phi \boldsymbol{zx}$ and $p\boldsymbol{z}$. This makes sure $q\phi \boldsymbol{zx}$ is close to $\mathcal{N}(0, 1)$ to allow us to sample it easily. After a VAE is trained, we could directly generate synthetic samples using its decoder.

As illustrated in Figure 3, we implement a VAE for generating solar and load daily synthetic data. We tried to keep the same convolutional and de-convolutional layers used in the DCGAN model for the encoder and decoder, however we ran into an issue known as model collapse where the decoder generates very limited diversity of samples. Instead we just implement a few dense layers for both encoder and decoder.

In contrast to the model structure of the DCGAN model, we did not use batch normalisation as empirically we found the model converges more quickly without it. We used ReLU activations for the decoder except the last output layer, in which we used a sigmoid function. ReLU and dropout functions are adopted for the encoder dense layers. RMSprop (Ruder, 2016) is used as the model optimiser for VAE instead of Adam.

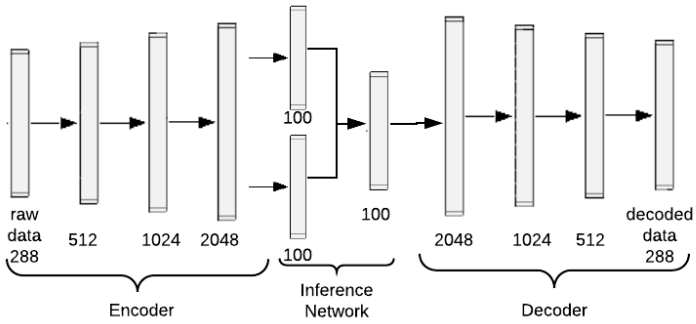

Figure 2. VAE Architecture used in this study. Numbers at the bottom of each layer indicate the outputs dimensions of each layer.

4. Results

4.1. Model Training

Both models were trained for 20000 training iterations, each using mini-batch of 32 samples. We use the same model structures for generating load and solar data, the whole training process using one type of data (solar or load) takes around one hour for the GAN model and approximately 50 minutes for the VAE model on a 2.9 GHz Intel Core i7 CPU.

Figure 4 demonstrates the batch training losses for each iteration when training our GAN and VAE. A few interesting points are summarised below based on the displayed training losses:

- The batch ELBOs of VAE seems to be fairly stable after approximately 1000 training steps with no further improvements.
- When training on solar data, the DCGAN's performance becomes consistent after around 5000 iterations. It is clear that the two functions reached a Nash equilibrium.
- On the other hand, when feeding in load data to our DCGAN, the losses are stable for a few thousand steps but after that the training losses destabilise and oscillate heavily. This oscillatory behaviour is intuitively explained in the study done by Mescheder et al. (2018) where they conclude that sometimes the cost functions in GAN will not converge using gradient descent. Despite the unstable training losses, we still find the quality of generated samples slowly improving throughout the training process.

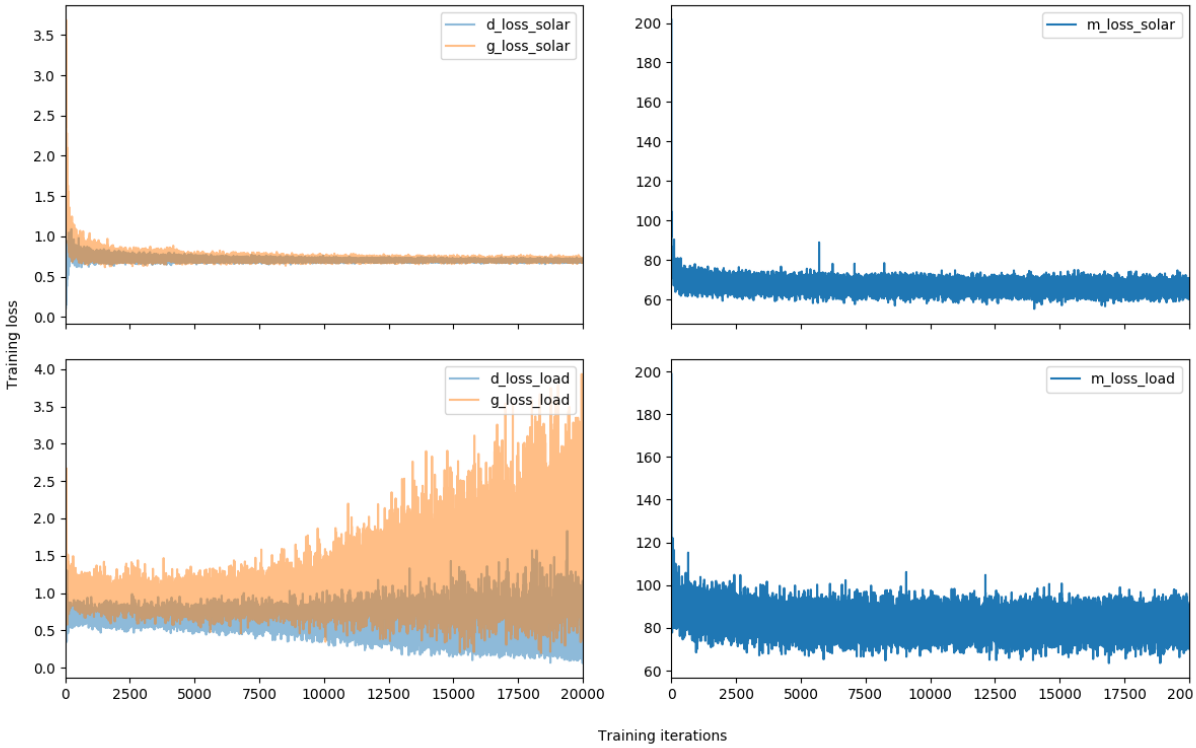

Figure 4: Training loss of GAN (left column) and VAE (right column) for generating solar (top row) and load (bottom row) scenarios. The plot label 'd_loss' indicates training loss for the discriminator (defined in Equation (1)), 'g_loss' is the loss for the generator (defined in Equation (2)) and 'm_loss' is the VAE training loss (defined in Equation (3)).

4.2. GAN vs VAE

Figure 5 and Figure 6 illustrate examples of generated solar and consumption scenarios for the two evaluated models, some real-time 5-minute data from the evaluation set are also plotted. It should be noted these samples are randomly drawn, not cherry picked. We denormalise the outputs from GAN and VAE by generating the same amount of samples as the test set and multiplying the model outputs by DC rated solar power and peak consumption power recorded in the test set.

A very rough manual grouping is done based on visualisations in Figure 5, where we put clear sky, partially cloudy and cloudy days in separate rows for easy visual comparison. Moreover, we added a column of the nearest training samples of the left neighbouring column to show that these evaluated generative models are not just memorising the training data.

As shown in Figure 5, the GAN model generates realistic samples regardless of the weather condition. On the other hand, the solar power scenarios generated by the VAE model cannot fully capture the bell-shaped curve on a clear sky day where we observe some unexpected noise in the middle of a day. Furthermore, the cloudy-day synthetic data from VAE omits power spikes present in the test data.

For generated load curves, it can be observed that the VAE model generates smoother plots compared to the GAN model. Compared to PV data, it is a bit more difficult to visually evaluate the

performance of these two models as every household can have very different load patterns. Moreover, these daily curves could come from any day of a year.

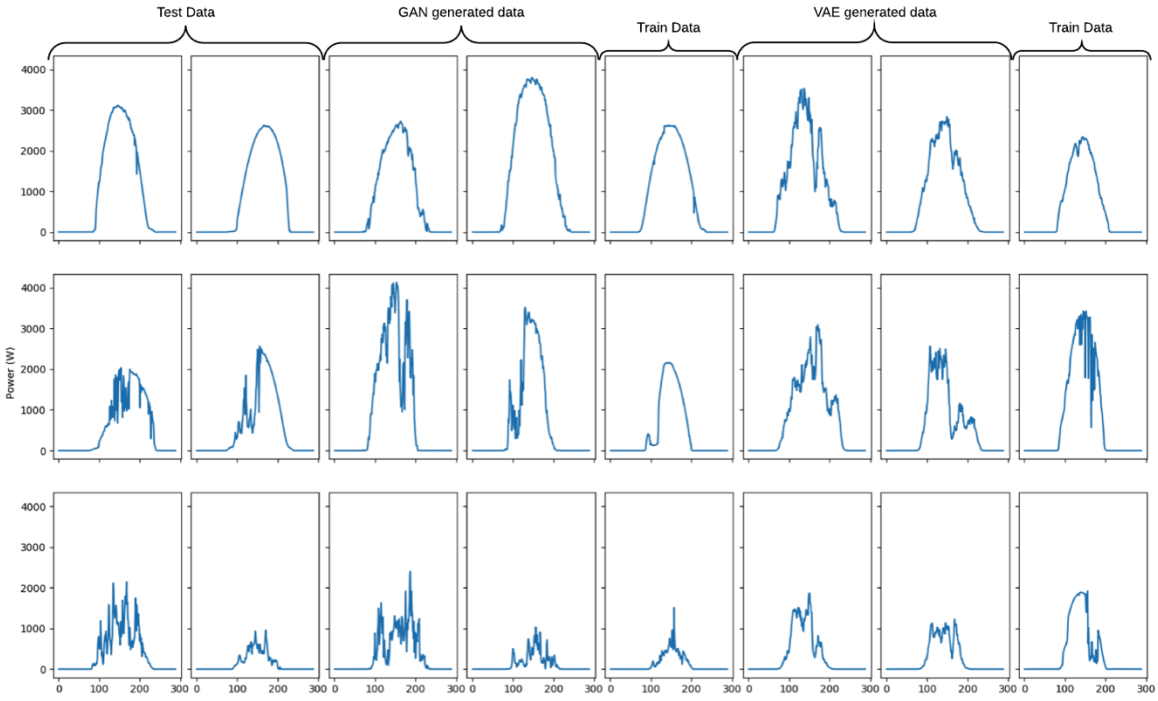

Figure 5: Generated 5-minute solar data using GAN and VAE.

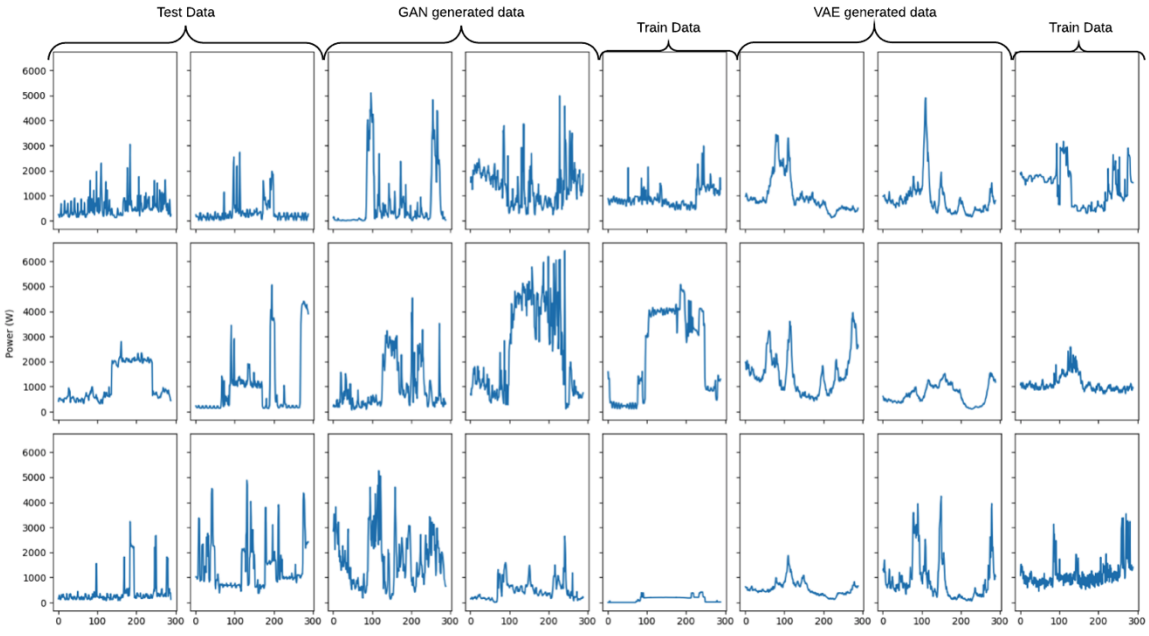

Figure 6: Generated 5-minute load data using GAN and VAE.

In Figure 7, we compare the Cumulative Distribution Functions (CDFs) of generated 5-minute power values and the 5-minute historical power data in the training and test set. It can be observed that the GAN model produces almost identical CDFs whereas there are some noticeable differences between the power distributions of VAE generated data and train/test data. Overall, judging by the sample qualities and CDFs the DCGAN model outperforms the VAE model although a small amount of additional computational costs is required.

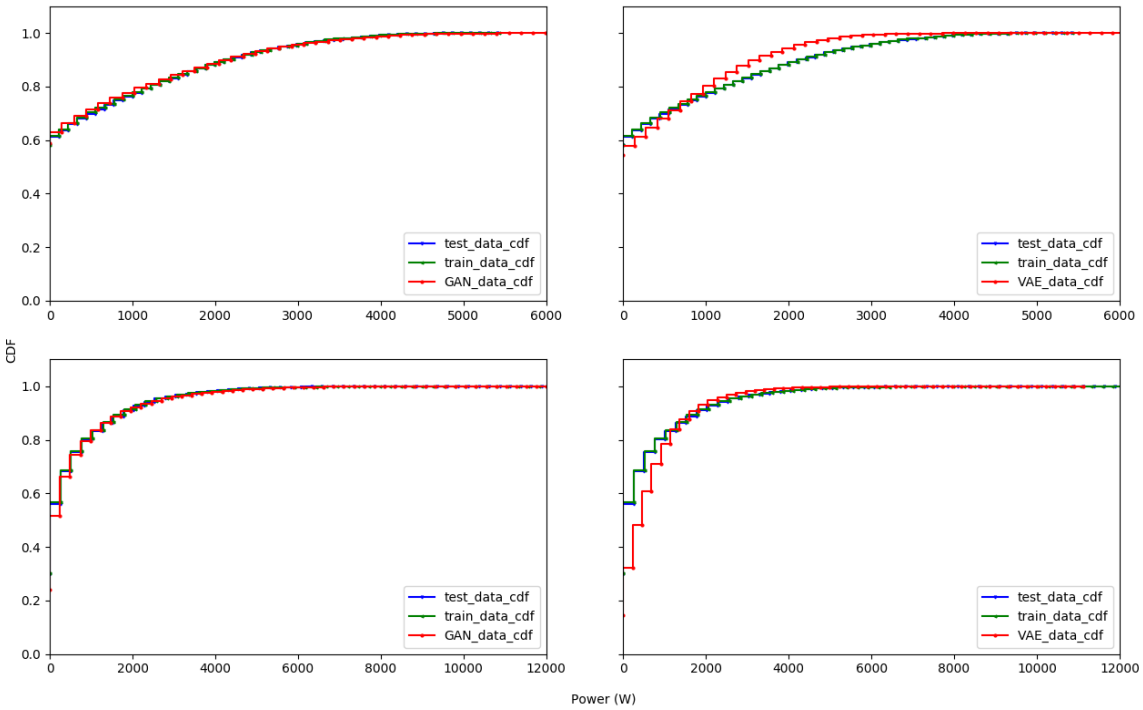

Figure 7: Cumulative Distribution Functions (CDFs) for generated and historical power data of PV (top row) and load (bottom row) for GAN (left column) and VAE (right column).

4.3. Case Study: Conditional GAN

Conditional GAN (CGAN) (Mirza & Osindero, 2014) allows us to provide additional information such as class labels or desired characteristics to the generator and discriminator. As a result, we can generate solar and consumption scenarios based on site specific conditions such as shading and user consumption patterns. In this study, we illustrate a case study on generating consumption data conditioned on user load patterns by applying a CGAN model.

We converted our DCGAN model to a conditional DCGAN by adding an input layer for both the generator and discriminator, therefore by feeding in load profile labels, the generator can generate corresponding scenarios and the discriminator receives more information to distinguish samples.

To add consumption pattern labels, we applied K-means clustering (MacQueen, 1967) to partition normalised consumption data into four clusters where cluster group numbers are adopted as consumption pattern labels. Then we feed in these labels to our CGAN model, same as the above approach for DCGAN, we keep 20% of total data as a test set to validate our generated samples.

Figure 8 illustrates samples generated by the CGAN, each column represents a separate clustered group. Although these synthetic profiles do have similar daily trends as the cluster centroids, we notice the qualities of these generated scenarios are not as good as the previous samples generated by DCGAN without any labels, where less noise occurs. We suspect the model is overfitting caused by insufficient data for each label: 1151 and 1915 samples for cluster 1 and 2,

3869 and 5519 samples for cluster 3 and 4. The CDFs for the train set, test set and generated samples are displayed in Figure 9. It can be observed that generated data CDF in cluster 4 where the most training samples are partitioned, has the closest distance to the train and test data. Overall, despite the limited amount of training data, the CGAN still captures most of the main time-correlation characteristics, variabilities and power distributions of the cluster groups.

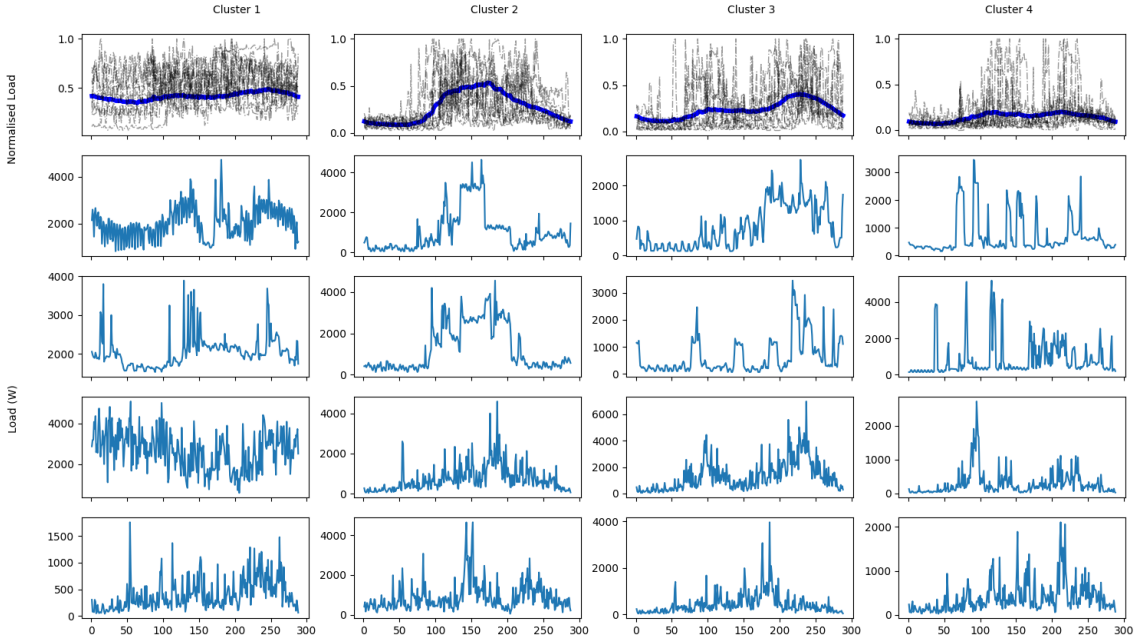

Figure 8: Samples generated by CGAN. The first row shows the normalised clustered profiles from the training set (black lines) and cluster centroids (blue lines) for each cluster. The second and third rows show randomly drawn load samples from the test set for each cluster and the fourth and fifth rows display randomly selected generated samples for each cluster.

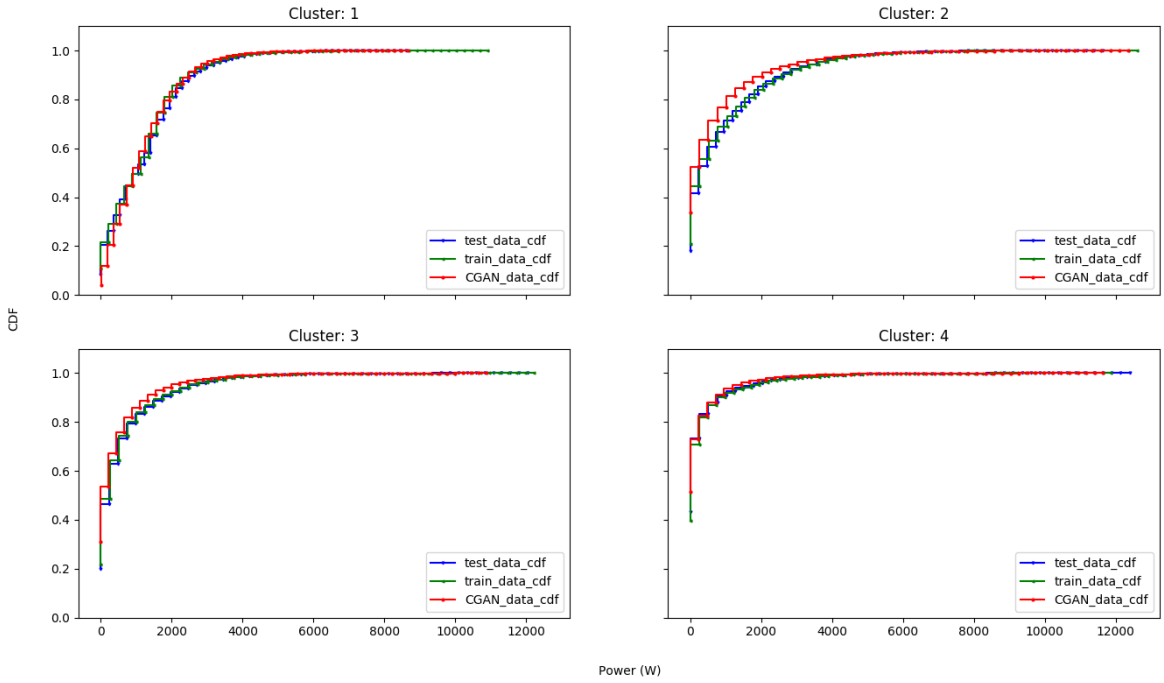

Figure 9: Cumulative Distribution Functions (CDFs) for CGAN generated and historical power consumption data, each subplot shows a different cluster label.

5. Conclusion and future work

In this study, a GAN-based tool is proposed to generate synthetic 5-minute solar generation and electricity consumption data. In contrast to other generative models, this approach is unsupervised, data-driven and simple to implement without requiring a complicated probabilistic model. More importantly, we demonstrate this tool can generate high quality generation and consumption scenarios within a reasonable amount of computational costs and with a small training set. Moreover, we illustrate that it is feasible to generate data conditioned on customer's consumption patterns.

It should be noted that even though we show that our GAN model outperforms the VAE model in terms of generated CDFs, it is still not concrete which approach is better as we used a simpler model architecture for VAE and currently there is no widely accepted metric to compare the performances between VAE and GAN. For future works, we believe it is important to develop a metric tailored for the end applications of the generated samples (e.g. power scheduling optimisation using solar and consumption data), given that most generative model metrics in the literature are still focused on images qualities and diversities. The training stability of GAN is also questionable due to the difficulty of achieving a Nash equilibrium and there are a lot of recently developed GAN models focusing on solving this problem. It would be interesting to see whether these new proposed models along with more training data could achieve a better performance on generating solar and consumption scenarios.

References

Arjovsky, M., Chintala, S. & Bottou, L., 2017. Wasserstein Generative Adversarial Network. International Conference on Machine Learning, pp.1–10. Available at: http://arxiv.org/abs/1701.07875.

Bloomberg NEF, 2018, New Energy Outlook 2018, accessed from https://bnef.turtl.co/story/neo2018?teaser=true on 1 October 2018.

Chandrashekeran, S., 2018, Smart electricity meters are here, but more is needed to make them useful to customers, accessed from http://theconversation.com/smart-electricity-meters-are-here-but-more-is-needed-to-make-them-useful-to-customers-92029 on 28 September 2018.

Chen, J., Kim, J.S. & Rabiti, C., 2017. Probabilistic analysis of hybrid energy systems using synthetic renewable and load data. Proceedings of the American Control Conference, pp.4723–4728.

Chen, Y. et al., 2017. Model-Free Renewable Scenario Generation Using Generative Adversarial Networks. , 33(3), pp.3265–3275. Available at: http://arxiv.org/abs/1707.09676.

Clean Energy Council (CEC), 2018, Clean Energy Australia Report 2018, accessed from https://www.cleanenergycouncil.org.au/policy-advocacy/reports/clean-energy-australia-report.html on 1 October 2018.

European Parliament, 2016, Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), Available at: https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:32016R0679.

Goodfellow, I., 2016. NIPS 2016 Tutorial: Generative Adversarial Networks. Available at: http://arxiv.org/abs/1701.00160.

Goodfellow, I.J. et al., 2014. Generative Adversarial Networks. , pp.1–9. Available at: http://arxiv.org/abs/1406.2661.

Ioffe, S. & Szegedy, C., 2015. Batch Normalization : Accelerating Deep Network Training by Reducing Internal Covariate Shift.

Iversen, E.B., Pinson, P. & Arduin, I., 2016. RESGen: Renewable Energy Scenario Generation Platform. 2016 IEEE Power & Energy Society General Meeting (PESGM), pp.1–5.

Kingma, D.P. & Welling, M., 2013. Auto-Encoding Variational Bayes. , (Ml), pp.1–14. Available at: http://arxiv.org/abs/1312.6114.

Kingma, D.P. and Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Maas, A.L., Hannun, A.Y. and Ng, A.Y., 2013, June. Rectifier nonlinearities improve neural network acoustic models. In Proc. icml (Vol. 30, No. 1, p. 3).

J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics, University of California Press, Berkeley, Calif., 1967, pp. 281-297.URL https://projecteuclid.org/euclid.bsmsp/1200512992.

Mescheder, L., Geiger, A. and Nowozin, S., 2018, July. Which Training Methods for GANs do actually Converge?. In International Conference on Machine Learning (pp. 3478-3487).

Mirza, M. & Osindero, S., 2014. Conditional Generative Adversarial Nets. , pp.1–7. Available at: http://arxiv.org/abs/1411.1784.

Nair, V. and Hinton, G.E., 2010. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th international conference on machine learning (ICML-10) (pp. 807-814).

Pillai, G.G., Putrus, G.A. & Pearsall, N.M., 2014. Generation of synthetic benchmark electrical load profiles using publicly available load and weather data. International Journal of Electrical Power and Energy Systems, 61, pp.1–10. Available at: http://dx.doi.org/10.1016/j.ijepes.2014.03.005.

Radford, A., Metz, L. & Chintala, S., 2016. Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks. , pp.1–16. Available at: https://arxiv.org/pdf/1511.06434.pdf.

Rezende, D.J., Mohamed, S. & Wierstra, D., 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. Available at: http://arxiv.org/abs/1401.4082.

Ruder, S., 2016. An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A. and Chen, X., 2016. Improved techniques for training gans. In Advances in Neural Information Processing Systems (pp. 2234-2242).

Suomalainen, K. et al., 2012. Synthetic wind speed scenarios including diurnal effects: Implications for wind power dimensioning. Energy, 37(1), pp.41–50. Available at: http://dx.doi.org/10.1016/j.energy.2011.08.001.

Tastu, J., Pinson, P., & Madsen, H. (2015). Space-time trajectories of wind power generation: Parameterized precision matrices under a Gaussian copula approach. In A. Antoniadis, J-M. Poggi, & X. Brossat (Eds.), Modeling and Stochastic Learning for Forecasting in High Dimensions (pp. 267-296). Springer. (Lecture Notes in Statistics; No. 217). DOI: 10.1007/978-3-319-18732-7_14.

Wang, T., Chiang, H. & Tanabe, R., 2016. Toward a Flexible Scenario Generation Tool for Stochastic Renewable Energy Analysis. Power Systems Computation Conference (PSCC) 2016.

View publication stats